www.ijecs.in
International Journal Of Engineering And Computer Science ISSN: 2319-7242
Volume 4 Issue 10 Oct 2015, Page No. 14653-14658

A High-Speed Montgomery Modular Multiplication Algorithm To Reduce The Energy Consumption Based On RSA Cryptosystems

S.Ashwini (Pg Scholar) [1] P.Thirapaiah Assistant Professor[2]

Department of ECE, Lords Institute of Engineering and Technology, Hyderabad, INDIA

Abstract

In the age of information, security issues play a crucial role. Security comes with three points' confidentiality, integrity and availability. The entire above said thing will come from an efficient cryptographic algorithm. Cryptosystem is usually achieved by repeated modular multiplications on large integers. To speed up the encryption decryption process, many high-speed Montgomery modular multiplication algorithms and hardware architectures employ carry save addition to avoid the carry propagation at each addition operation of the add-shift loop. In this paper, we propose an energy-efficient algorithm and its corresponding architecture to not only reduce the energy consumption but also further enhance the throughput of Montgomery modular multipliers. The proposed architecture is capable of bypassing the superfluous carry-save addition and register write operations, leading to less energy consumption and higher throughput. In addition, we also modify the barrel register full adder (BRFA) so that the gated clock design technique can be applied to significantly reduce the energy consumption of storage elements in BRFA. Experimental results show that the proposed approaches can achieve up to 60% energy saving and 24.6% throughput improvement for 1024-bit Montgomery multiplier.

KEYWORDS: Carry-save addition, Energy-efficient architecture gated clock, Montgomery modular multiplier, RIVEST, SHAMIR, and ADLEMAN (RSA) cryptosystem

1. INTRODUCTION

The growth of the Internet and electronic commerce has brought to the forefront the issue of privacy in electronic communication. Large volumes of personal and sensitive information are electronically transmitted and stored every day. What a guarantee does one have that a message sent to another person is not intercepted and read without their knowledge or consent? Tools to ensure the privacy and confidentiality of paper-based communication have existed for a long time. Similar tools exist in the electronic communications arena. Encryption is the standard method for making a communication private. Anyone wanting to send a private message to another user encrypts (enciphers) the message before transmitting it. Only the intended recipient knows how to correctly decrypt (decipher) the message. Anyone who was "eavesdropping" on the communication would only see the encrypted message. Because they would not know how to decrypt it successfully, the message would make no sense to them. As such, privacy can be ensured in electronic communication.

Many systems utilize public-key cryptography to provide such security services, and Rivest, Shamir, and Adleman (RSA) is one of the most widely adopted public key algorithms at present. However, RSA requires repeated modular multiplications to accomplish the computation of modular exponentiation, and the size of modulus is generally at least 1024 bits for long-term security. Therefore, high data throughput rates without hardware acceleration are difficult to be achieved. Additionally, security requirements are increasingly important for private data transmission through mobile devices with Internet access, such as smart phones and notebook computers, which require an energy-efficient cryptosystem due to their limitation of battery power.

A popular approach to implement modular multiplication in hardware circuits is based on the Montgomery modular multiplication algorithm, which replaces the trial division by modulus with a series of addition and shift operations, and thus its critical operation is basically a three-operand addition with an iteration loop. Unfortunately, time-consuming carry propagations from the addition of long operands seriously influence the performance of the RSA cryptosystems. Accordingly, several approaches toward avoiding long carry propagation during the addition operation have been proposed to achieve a significant speed up of Montgomery modular multiplication. These approaches can be roughly classified into two categories relying on the representation of intermediate results of modular multiplication in the exponentiation.

2. RSA ALGORTIHM

2.1 History of RSA Algorithm

In 1978, Ron Rivest, Adi Shamir, and Leonard Adleman introduced a cryptographic algorithm, which was essentially to replace the less secure National Bureau of Standards (NBS) algorithm. Most importantly, RSA implements a public-key cryptosystem, as well as digital signatures. RSA is motivated by the published works of Diffie and Hellman from several years before, who described the idea of such an algorithm, but never truly developed it. Introduced at the time when the era of electronic email was expected to soon arise, RSA implemented two important ideas: 1. Public-key encryption. This idea omits the need for a "courier" to deliver keys to recipients over another secure channel before transmitting the originally-intended message. In RSA, encryption keys are public, while the decryption keys are not, so only the person with the correct decryption key can decipher an encrypted message. Everyone has their own encryption and decryption keys. The keys must be made in such a way that the decryption key may not be easily deduced from the public encryption key. 2. Digital signatures. The receiver may need to verify that a transmitted message actually originated from the sender (signature), and didn't just come from there (authentication). This is done using the sender's decryption key, and the signature can later be verified by anyone, using the corresponding public encryption key. Signatures therefore cannot be forged. Also, no signer can later deny having signed the message.

2.2 Basics of cryptography

Cryptography is the study of building ciphers to ensure that the information are hidden from unauthorized access (confidentiality) and protected from unauthorized changes (integrity). In other words it is the process of converting the plaintext into cipher text and vice versa. The above things are done by using some cryptographic algorithm and secret keys. The original message that a sender is going to send is known as plaintext and the message that is sent through the channel is known as cipher text. In classical cryptography, the same secret key is used for encryption and decryption known as symmetric key cryptography. On the other hand in modern cryptography different secret keys are used for encryption and decryption known as asymmetric key cryptography.

3. ARCHITECTURE OF EXISTING SYSTEM

The block diagram of existing system MMM42 multiplier Fig.1. In addition to MBRFA and LU, the main component is the four-to-two CSA architecture. The CSA structure can be combined with other techniques and architectures to further improve the performance of Montgomery multipliers. However, these designs probably cause a large increase in hardware complexity and power consumption. So we will propose a new Montgomery algorithm.

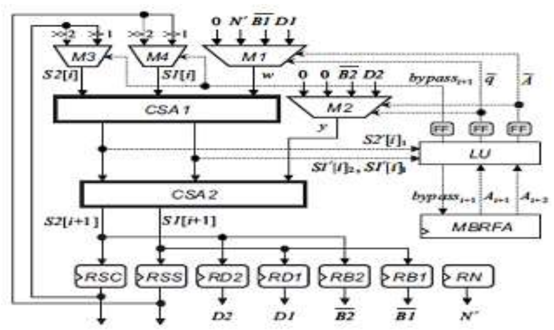
0 N' $\overline{B1}$ D1
M3
M4
M1
S2[i]
S1[i]
w
0 0 $\overline{B2}$ D2
bypass$_{i+1}$ $\overline{q}$ $\overline{A}$
CSA 1
M2
y
FF FF FF
S2'[i]$_1$
LU
S1'[i]$_2$, S1'[i]$_1$
CSA 2
bypass$_{i+1}$ A_{i+1} A_{i+2}
MBRFA
S2[i+1]
S1[i+1]
RSC RSS RD2 RD1 RB2 RB1 RN
D2 D1 $\overline{B2}$ $\overline{B1}$ N'

Figure 1: Block diagram of existing system MMM42 Multiplier

4. ENERGY-EFFICIENT MONTGOMERY MODULAR MULTIPLICATION ALGORITHM

In the following sections, we will propose a new Montgomery algorithm to bypass the superfluous operations without increasing the critical path delay of Montgomery multipliers using four-to-two RCA architecture is proposed to reduce the energy consumption and enhance the throughput via bypassing the superfluous operation.

Algorithm MMM42: 4-to-2 CSA Modified Montgomery Multiplication

```
Inputs: $A1, A2, B1, B2, N(modulus)$
Outputs: $S1[K], S2[K]$
1. $(D1, D2) = B1 + B2 + N + 0;$
2. $S1[0] = 0; S2[0] = 0;$
3. $for\ i = 0\ to\ k - 1\ \{$
4. $q_i = \big(S1[i]_0 + S2[i]_0 + A_i \times (B1_0 + B2_0)\big) mod\ 2;$
5. $if(A_i = 0\ and\ q_i = 0)$
6. $(S1[i+1], S2[i+1]) = (S1[i] + S2[i] + 0 + 0)/2$
7. $else\ if(A_i = 0\ and\ q_i = 0)$
8. $(S1[i+1], S2[i+1]) = (S1[i] + S2[i] + N + 0)/2;$
9. $else\ if\ (A_i = 1\ and\ q_i = 0)$
10. $(S1[i+1], S2[i+1]) = (S1[i] + S2[i] + B1 + B2)/2$
11. $else\ if(A_i = 1\ and\ q_i = 1)$
12. $(S1[i+1], S2[i+1]) = (S1[i] + S2[i] + D1 + D2)/2$
13. }
14. Return $S1[K], S2[K]$
```

As shown in Algorithm MMM42, steps 1 and 2 for producing the carry values (B1, B2) and (D1, D2) are first performed to easily realize the subsequent quotient look-ahead. Because the qi+1, Ai+1, qi+2, and Ai+2 must be generated at the i th iteration, the iterative index i of Montgomery modular multiplication will start from −1 instead of 0 and the corresponding initial values of ˜ q and ˜A must be set to 0. Additionally, the original for loop in Algorithm MMM42 is replaced with the while loop to bypass some superfluous iterations when bypassi+1 = 1.

Note that the ending number of iterations in Algorithm MMM42 is changed to k + 2 instead of k−1due to the following reasons. First, the convergence range of S in Algorithm MM falls in the range of [0, 2N), thus an additional operation S = S − N is required to keep the range of output S in [0, N) if S ≥ N. We employ the notion of Walter's approach [24] to remove the time-consuming subtraction and maintain the range of input operands A, B, and output S within [0, 2N) through increasing two extra iterations and extending (A1, A2) and (B1, B2) to k + 2 bits. Because (A1, A2) and (B1, B2) are unsigned numbers, two dummy zeros are directly inserted in front of the MSB of these operands. Second, the carry values (B1, B2) = 2B1 + 2B2 is computed at the beginning of Algorithm MMM42.

Therefore, an extra iteration for computing division by two is necessary to ensure the correctness of Montgomery modular multiplication. In this manner, the output (S1, S2) can be utilized as an input in the consecutive modular multiplication without the additional Subtraction while carrying out the modular exponentiation. In the while loop, steps 6– 13 will be performed in a four-to-two RCA architecture with two 4-to-1 multiplexers. Computations of Ai+1 and Ai+2 in step 14 can be executed in parallel with the four-to-two RCA architecture. In addition, computations of qi+1, qi+2, and bypassi+1 in step 14 can be carried out after S1_[i]1, S2_[i]1, and S1_[i]2 have been produced by

RCA1 (i.e., the upper FAs of the four-to-two RCAS architecture) Subsequently, steps 15–21 except step 17 can be performed after step 14 has been completed.

5. ENERGY-EFFICIENT HARDWARE DESIGN

In this section, the fundamental hardware architecture of Algorithm MMM42, denoted as MMM42 multiplier, is first described. Next, the gated clock design technique is applied to the proposed MMM42 multiplier to further reduce the energy consumption.

5.1 Modified Barrel Register Full Adder (MBRFA) and Look-Ahead Unit (LU)

As shown in steps 15–21 of Algorithm MMM42, ˜ q and ˜A must be generated and stored at the i th iteration according to bypassi+1 signal so that they can be used to select the correct input operands of four-to-two carry save addition at the next clock cycle. However, Ai+1 and Ai+2 are needed to reduce ˜A . Therefore, the BRFA shown in Fig. 1 must be modified to be able to generate the values of Ai+1 and Ai+2 at the same iteration. The modified BRFA (denoted as MBRFA) depicted in Fig. 2 employs two shift registers RA1 and RA2 with two full adders to generate Ai+1 sand Ai+2 at the same clock cycle.

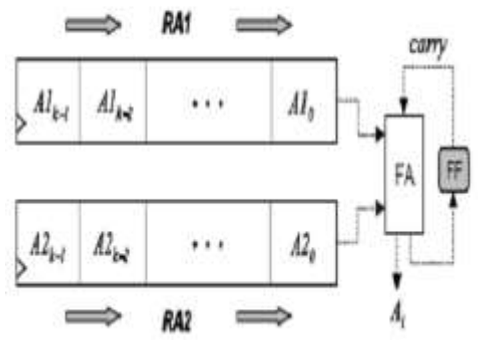

Figure 2: BRFA

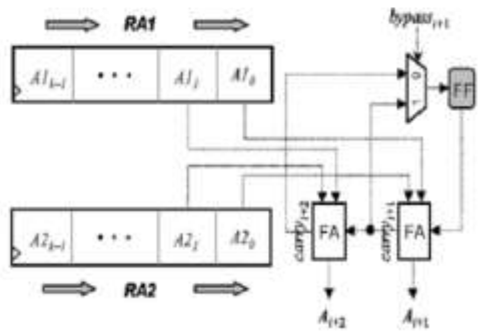

Figure 3: MBRFA

After Ai+1 and Ai+2 are obtained, a LU depicted in Fig. 5 is developed to generate the bypassi+1 signal, ˜ q, and ˜A. The LU consists of an XOR gate, an NOR gate, and two 2-to- 1 multiplexers. It produces the qi+1, qi+2, and bypassi+1 signal respectively.

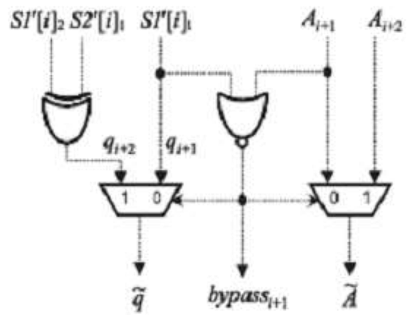

Figure 4: Look-Ahead Unit (LU)

Afterward q and ˜A are selected respectively. Additionally, the bypassi+1 signal must be sent back to the MBRFA to determine the carry value that should be stored to FF and the bit number that registers RA1 and RA2 must be right shifted. If bypassi+1 = 1, carryi+2 is stored to FF and registers RA1 and RA2 are right shifted by two bit positions. Otherwise, carryi+1 is stored to FF and registers RA1 and RA2 are right shifted by one bit position.

5.2 Architecture of MMM42 Multiplier

The block diagram of proposed MMM42 multiplier Figure 5, in addition to MBRFA and LU, the main component is the four-to-two RCA architecture. It is first used to pre-compute the four-to-two RCA value in step 1 and step 2 of Algorithm MMM42 through setting ˜A, ˜ q, and registers to proper values. The produced (B1, B2) are stored to registers RB1 and RB2, and (D1, D2) are stored to registers RD1 and

RD2, respectively. When carrying out the computation of iteration i for −1 ≤ i ≤ k + 2, the four to- two RCA architecture selects the proper w and y through multiplexers M1 and M2 by using ˜A and ˜ q produced at the previous clock cycle. In addition, the intermediate values S1_[i]1, S2_[i]1, and S1_[i]2 produced by RCA1 of the four to- two RCA architecture together with the Ai+1 and Ai+2produced by MBRFA are sent to LU to generate the bypassi+1 signal and the values of ˜ q and ˜A for the next clock cycle.

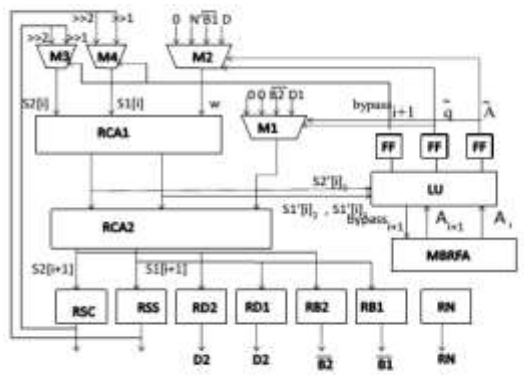

Figure 5: Block diagram of proposed MMM42 Multiplier.

The outputs S1[i+1] and S2[i+1] of four-to-two RCA architecture must be divided by two as shown in the steps 6–13 of Algorithm . When bypassi+1 = 1, however, the outputs of four-to-two RCA architecture must be divided by four (i.e., right shifted by two bit positions). One possible method to perform the extra right-shift one-bit operation (i.e., step 17 of Algorithm MMM42) is inserting two extra 2-to-1 multiplexers M3 and M4 with bypassi+1 as select signal to the front of RSCj and RSSj , respectively. If bypassi+1 = 1, M3 and M4 will select S1[i+1] j+1 and S2[i+1] j+1, respectively. Otherwise, S1[i+1] j and S2[i+1] j will be selected through M3 and M4. Instead of the above method, we store the bypassi+1 signal to FF and insert multiplexers M3 and M4 at the front of RCA1 as shown in Fig. 4. As a result, the right-shift one-bit or two-bit operation will be performed at the next clock cycle according to the bypassi+1 signal stored in FF. In this manner, the glitches of bypassi+1 signal, which probably causes unnecessary dynamic power consumption will be blocked by FF and not be propagated to M3, M4, and RCA. Furthermore, because the additional M3 and M4 operate in parallel with M1, the critical path delay is not affected by such an arrangement.

5.3 Gated Clock Design Technique

Lessening the power/energy consumption of registers will also lower that of MMM42 multiplier since several registers are required in the multiplier to store the inputs, outputs to achieve higher performance. We can see that the registers RB1, RB2, RD1, RD2, and RN in Fig. 5.rarely load new value when performing Montgomery multiplication operations.

6. RESULTS

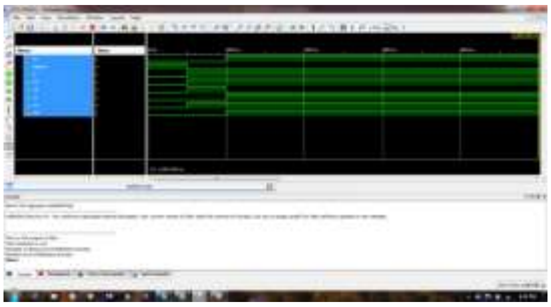

Figure 6: Look Ahead Unit

Figure 7: Barrel Register Full Adder

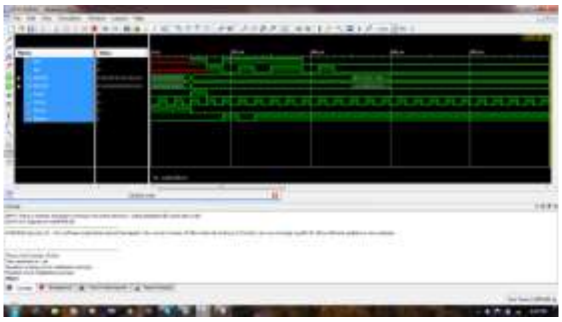

Figure 8: Modular Multiplier Test

DOI: 10.18535/ijecs/v4i10.19

Figure 9: Modified Rivest Shamir and Adleman

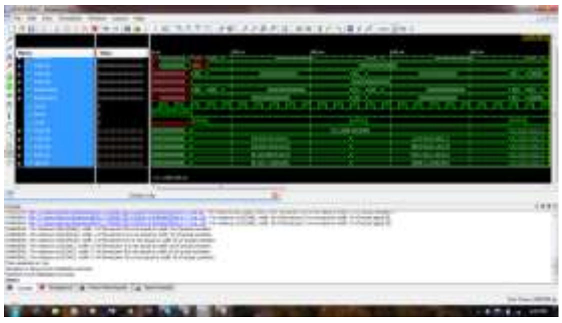

Figure 10: Modular Multiplier Exponentiation Test

7. CONCLUSION

More registers and higher energy consumption were introduced into the high-speed Montgomery modular multipliers, which speed up the decryption/encryption process by maintaining all inputs and outputs of the modular multiplication in a redundant carry save format. This paper presented an efficient algorithm and its corresponding architecture to reduce the energy consumption and enhance the throughput of Montgomery modular multipliers simultaneously. Moreover, we modified the structure of BRFA and adopted the gated clock design technique to further reduce the energy consumption of Montgomery modular multipliers. Experimental results showed that the proposed approaches are indeed capable of reducing the energy consumption of the Montgomery multipliers. In the future, we will try to heighten the occurring probability of superfluous operation bypassing to further reduce the energy consumption and enhance the throughput of modular multiplication.

REFERENCES

[1] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signature and public-key cryptosystems,”Commun. ACM, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[2] P. L. Montgomery, “Modular multiplication without trial division,”Math. Comput., vol. 44, no. 170, pp. 519–521, Apr. 1985.

[3] C. K. Koc, T. Acar, and B. S. Kaliski, “Analyzing and comparing Montgomery multiplication algorithms,”IEEE Micro, vol. 16, no. 3, pp. 26–33, Jun. 1996.

[4] Y. S. Kim, W. S. Kang, and J. R. Choi, “Implementation of 1024-bit modular processor for RSA cryptosystem,” inProc. IEEE Asia-Pacific Conf., Aug. 2000, pp. 187–190.

[5] V. Bunimov, M. Schimmler, and B. Tolg, “A complexity-effective version of Montgomery’s algorithm,” in Proc. Workshop Complexity Effect. Designs, May 2002, pp. 1–7.

[6] A. Cilardo, A. Mazzeo, N. Mazzocca, and L. Romano, “A novel unified architecture for public-key cryptography,” inProc. Design, Autom. Test Eur. Conf. Exhibit., Mar. 2005, pp. 52–57.

[7] Z. B. Hu, R. M. A. Shboul, and V. P. Shirochin, “An efficient architecture of 1024-bits Cryptoprocessor for RSA cryptosystem based on modified Montgomery’s algorithm,” inProc. 4th IEEE Int. Workshop Intell Data Acquisit. Adv. Comput. Syst., Sep. 2007, pp. 643–646.

[8] C. McIvor, M. McLoone, and J. V. McCanny, “Modified Montgomery modular multiplication and RSA exponentiation techniques,”IEE Proc.-Comput. Digit. Tech., vol. 151, no. 6, pp. 402–408, Nov. 2004.

[9] K. Manochehri and S. Pourmozafari, “Fast Montgomery modular multiplication by pipelined CSA architecture,” in Proc. IEEE Int. Conf. Microelectron., Dec. 2004, pp. 144–147.